Design Document

Bruce Cain

August 2018

1 Decode

Decode begins by reading in a pcap file form the command line and attempts to open it. If it fails it will exit the program, if not then it continues to start reading in the data. The goal of decode was to take the pcap pull out the Zerg header and print out the data found inside. Using structs for each header and the first to be read in is the pcap file header. Things noted is the magic number to determine if its valid, big endian, or little endian. Next to check is the major and minor version since the only thing support is 2.4. Then next is to check the link layer to make sure its a Ethernet packet.

Next is the pcap packet header where the field to take notice is the size. The size is not checked until the IP header is read in and check its version to determine the size. If its IPv4 the IHL must be checked to verify its size. Once that is read in the size can be verified. Once that is completed next is the zerg payload.

Depending on the type provided in the Zerg header is how we move to the next header to parse. If its invalid the program exits. If its type is 0 it prints out the message. Type 1 will pull out the status header and use the message function to print out the name as well. Type 2 prints out the commands depending on the command type in the header. Type 3 prints out the GPS header for a zerg. This continues till the program cannot find anymore pcap file headers.

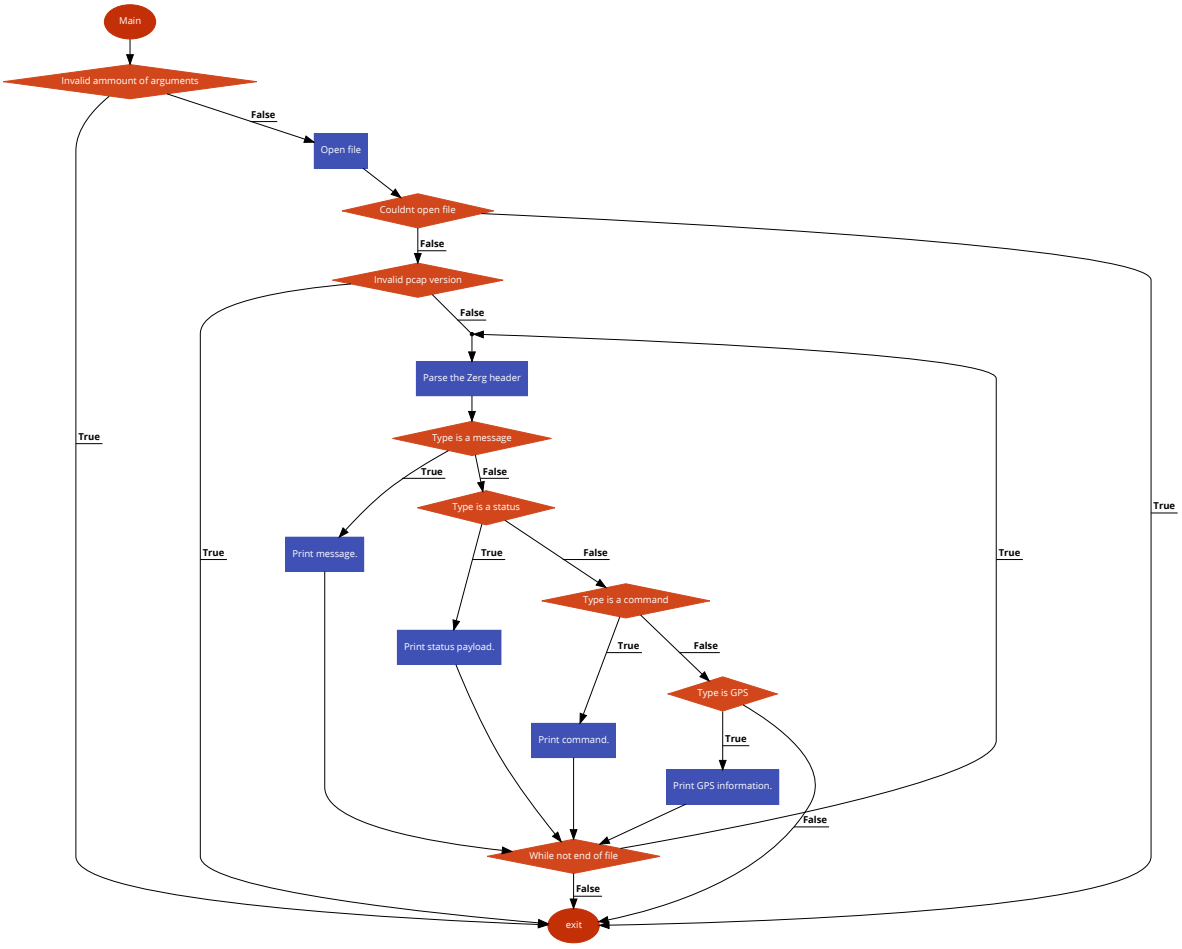

Figure 1: Decode Flow Diagram

2 Encode

Encode begins by checking if it can open both input and output files given as arguments form the terminal. Once that is done using the inFile, input file, to check for the zerg banner that decode puts between every packet. If found it will check for the main header fields, if any are out of order or not found it will exit.

Once parsed using the type field to do the reverse of decode. Using the structs it will populate them with the data found in the inFile. Once all validation is done a function is called to start writing all the headers to the outFile, output file, in the correct order. This will continue till either a error or another zerg banner is found. If neight happen the program exits normally.

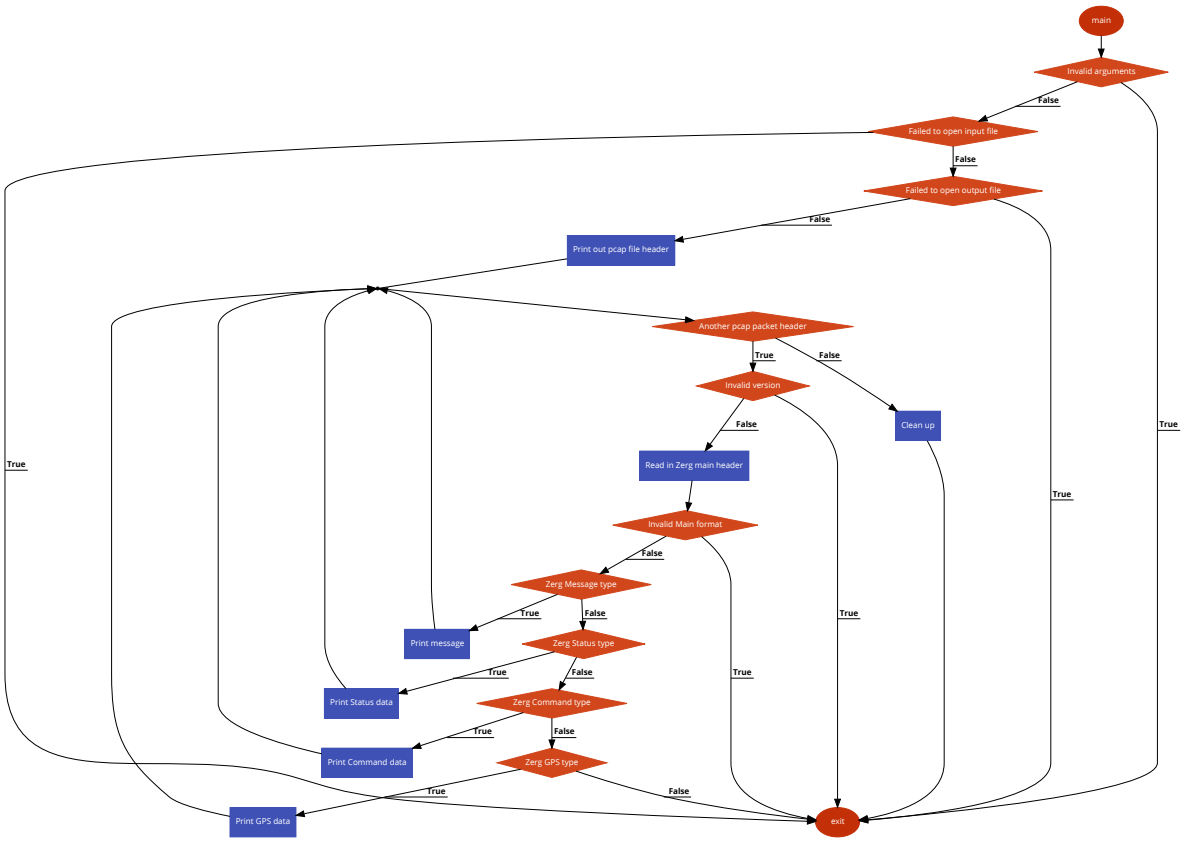

Figure 2: Encode Flow Diagram